Project Report CS-433

Deprelle Théo Prasopoulos Konstantinos Dona Ergys
Department of Computer Science, EPFL Lausanne, Switzerland

***Abstract*—The aim of this project is to use various Machine Learning techniques in order to recreate the process of "discovering" the Higgs particle, by using accelerator data provided by the Large Hadron Collider at CERN. Specifically, the following methods are used: linear regression with (stochastic) gradient descent, least squares, ridge regression, (regularised) logistic regression with (stochastic) gradient descent. We evaluate their performance and argue about how we can deal with the abnormalities found in the data-set.**

I. The data set

The data-set consists of input data containing 30 features. Their distribution is diverse and there is an important range difference among the values of the input vectors. Because of this, data pre-processing was an important part due to the need to normalise the "distance" among the features so that they can be treated equally by the regression algorithms.

In order to locally evaluate our models we split the data-set into 80% training data and 20% test data.

II. Data Pre-Processing

To begin with, we observed that the input data vectors have some missing values, represented by -999. Since we are going to do data standardisation (which implies that the mean of each feature will be zero), we ignored those values in the standardisation process and set those values to zero afterward.

To better fit the model, we enriched the input vectors by adding polynomial terms, up to a particular degree M, as shown below:

$$\mathbf{x}_n = [x_{n1}, \ldots, x_{nd}] \rightarrow \left[1, x_{n1}, \ldots, x_{nd}^M, x_{n2}, \ldots\right]$$

Finally, we normalised the features, by subtracting the mean and dividing by the variance:

$$\mathbf{x}_k = \frac{\mathbf{x}_k - \bar{\mathbf{x}}_k}{\sigma}, \quad \text{for each feature column } \mathbf{x}_k$$

The normalisation step had to be carried out after adding the polynomial terms, because the addition of these terms would "de-normalise" the features.

III. Baseline using Ridge Regression

Ridge Regression was selected to produce the baseline for the project for the following reasons: a) thanks to RR's regularization, the chance of overfitting is reduced b) RR is the Least Squares method improved which in turn is the explicit solution to linear regression; therefore RR was picked as the most general of the linear regression methods.

The baseline was obtained using $\lambda = 0.01$ and keeping the data in their original state. The model yielded 74.352% test accuracy.

By removing all 11 out of 30 features that contain N/A data, local accuracy fell to 72.25%. By removing all samples that contain N/A data (about 3/4) local and online accuracy fell to 72.12% and 70.98% respectively. The loss in accuracy can be attributed to the reduction of available training data. In the second case, there was an additional issue: testing data (from test.csv) could not go through the same sample-removal process since there is a specific amount of labels that need to be produced for submission. Since these changes reduce accuracy, they won't be used in the subsequent steps.

Lambda tuning is an obvious benefit to be had, especially when dealing with polynomial features. Tuning was implemented by running the Ridge Regression with a logarithmic range (from $\log(-5)$ to $\log(0)$) of lambdas and selecting that which minimises the local testing error. This way, accuracy was marginally increased to 74.4 locally and 74.437 online.

The next step was the augmentation of the training set. Using 3rd degree polynomials, local and online accuracy increased to 76.70% and 77.77%, respectively. At this point, increasing the degree results in lower accuracy. It is clear that the NA values must be dealt with.

To this end, we have applied the preprocessing describe in II. Now, a degree 3 feature set yields 78.7% while a degree 7 returns 79.58%.

Using the cross product feature generation described in section III results in a marginally improved accuracy of 79.88% at the expense of training time (465 features)

The above results are summarised in table I below. Each row represents one change in the technique - whether that concerns pre-processing or hyper-parameter tuning. Table II includes results from other linear methods which are generaly inferior to Ridge Regression.

Method	Accuracy (%)
Baseline ($\lambda = 0.01$)	74.32
"Bad" feature removal	72.25
"Bad" sample removal	71.12
λ–tuning (log range)	74.40
Using 3rd degree polys	76.70
Cleaning and standardising, 3rd deg.	78.70
Using 7th degree polys	79.58
Cross product feature set	79.88

TABLE I
Summarised results for RR (accuracy)

IV. Logistic Regression

The next step was to try and train our model using Logistic Regression. The loss formula of the model is:

$$\mathcal{L}(\mathbf{w}) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_n \log \sigma(\mathbf{x}_n^\top \mathbf{w}) + (1-y_n)\log\left(1-\sigma(\mathbf{x}_n^\top \mathbf{w})\right)\right] + \mathrm{R}(\mathbf{w})$$

where:

$$\mathrm{R}(\mathbf{w}) = \frac{\lambda}{2N}\sum_{j=1}^{d} w_j^2$$

$$\sigma(\mathbf{x}_n^\top \mathbf{w}) = \frac{1}{1+e^{-\mathbf{x}_n^\top \mathbf{w}}}$$

This loss is taking into account the regularisation quantity $\mathrm{R}(\mathbf{w})$, used to reduce over-fitting.

The gradient of the loss function, with respect to element w_0, is given by:

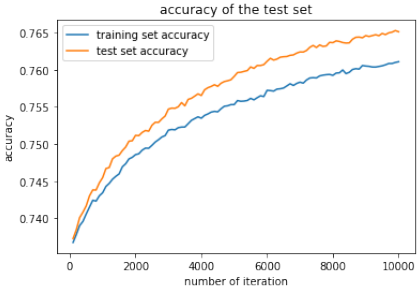
$$\frac{\partial \mathcal{L}(\mathbf{w})}{\partial w_0} = \frac{1}{N}\sum_{n=1}^{N}(\sigma(\mathbf{x}_n^\top \mathbf{w}) - y_n)x_{n0}$$

and with respect to w_k, $k > 1$, is given by:

$$\frac{\partial \mathcal{L}(\mathbf{w})}{\partial w_k} = \frac{1}{N}\sum_{n=1}^{N}(\sigma(\mathbf{x}_n^\top \mathbf{w}) - y_n)x_{nk} + \frac{\lambda}{N}w_k$$

The above method uses stochastic gradient descent with step γ and regularisation factor λ. Our goal was to maximise the accuracy of our test set before trying to predict the competition set.

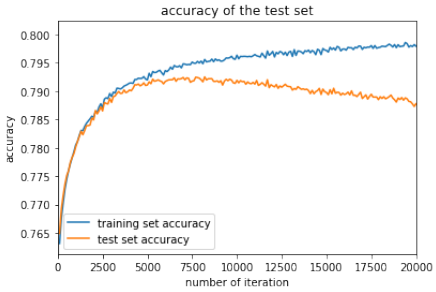
For our first training, we used $\lambda = 0, \gamma = 0.1, M = 9$ (i.e. 271 features) and a batch size of 32 elements. The loss and the accuracy are shown in the figure below.



Because the training set accuracy was lower than the test set accuracy our method was under-fitting. To solve this problem we decided to increase the batch size and the number of features by modifying our initial polynomial approach. This time we applied the following transformation:

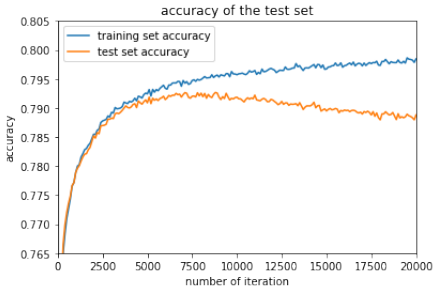
$$\mathbf{x}_n = [x_{n1}, \ldots, x_{nd}] \rightarrow \left[1, x_{n1}, x_{n1}^2, x_{n1}x_{n2}, \ldots, x_{n1}x_{nd}, x_{n2}, x_{n2}^2, x_{n2}x_{n3}, \ldots, x_{nd}^2\right]$$

This transformation allows us to have 465 features. For the training we have increased the number of iterations and we kept the same γ, λ and batch size. The loss and the accuracy are shown in the figure below.



With this approach, we are clearly over-fitting. This is mainly because λ was set to 0. A third training attempt was made, this time with $\lambda = 10^{-6}$.

As shown below, regularisation didn't help with overcoming the over-fitting problem.



Best of Ridge vs Logistic	Accuracy (%)
Ridge regression	
Standardized, 3rd degree, lambda tuning	78.70
Standardized, 7th degree, lambda tuning	79.58
Std, lamda tuning, Cross product feature set	79.88
Logistic regression	
$\lambda = 0, D = 9, \gamma = 0.1$ Batch size = 32, 10K it	76.58
$\lambda = 0, D = 9, \gamma = 0.1$ Batch size = 50K, 50K it	76.03
$\lambda = 0\gamma = 0.1$ Cross procuct, Batch size = 50K, 20K it	78.82
$\lambda = 1^{-3}, \gamma = 0.1$ Cross procuct, Batch size = 50K, 20K it	78.82
$\lambda = 1^{-6}, \gamma = 0.1$ Cross procuct, Batch size = 50K, 20K it	78.83

TABLE II
SUMMARISED RESULTS (ACCURACY)

V. Conclusion

With this project we have worked on different approaches of machine learning on a classification problem. Our best result was obtain using Regularized Logistic Regression with degree 5 polynomials. Still, accuracy has not significantly exceeded that of ridge regression because we did not find a way to reduce the over-fitting in Logistic Regression. To improve our result we should look deeper into the data set to enhance our data pre-processing pipeline.